PCT/FR03/03309
Rec'd PCT/PTO 12 NOV 2004
10/534855
MAILED 26 JAN 2004
WIPO PCT

INPI
INSTITUT NATIONAL DE LA PROPRIETE INDUSTRIELLE

BREVET D'INVENTION

CERTIFICAT D'UTILITÉ - CERTIFICAT D'ADDITION

COPIE OFFICIELLE

Le Directeur général de l'Institut national de la propriété industrielle certifie que le document ci-annexé est la copie certifiée conforme d'une demande de titre de propriété industrielle déposée à l'Institut.

Fait à Paris, le 05 NOV. 2003

Pour le Directeur général de l'Institut
national de la propriété industrielle
Le Chef du Département des brevets

Martine PLANCHE

)CUMENT DE PRIORITÉ

RÉSENTÉ OU TRANSMIS
CONFORMÉMENT À LA
RÈGLE 17.1.a) OU b)

INSTITUT NATIONAL DE LA PROPRIETE INDUSTRIELLE
SIEGE
26 bis, rue de Saint Petersbourg
75800 PARIS cedex 08
Téléphone : 33 (0)1 53 04 53 04
Télécopie : 33 (0)1 53 04 45 23
www.inpi.fr

INPI
INSTITUT NATIONAL DE LA PROPRIETE INDUSTRIELLE
26 bis, rue de Saint Pétersbourg
75800 Paris Cedex 08
Téléphone : 33 (1) 53 04 53 04 Télécopie : 33 (1) 42 94 86 54

BREVET D'INVENTION
CERTIFICAT D'UTILITÉ

Code de la propriété intellectuelle - Livre VI

cerfa N° 11354*03

REQUÊTE EN DÉLIVRANCE
page 1/2

BR1

Cet imprimé est à remplir lisiblement à l'encre noire DB 540 c 12 / 210502

Réservé à l'INPI

REMISE DES PIÈCES
DATE 14 NOV 2002
LIEU 75 INPI PARIS B
N° D'ENREGISTREMENT NATIONAL ATTRIBUÉ PAR L'INPI 0214279
DATE DE DÉPÔT ATTRIBUÉE PAR L'INPI 14 NOV. 2002

Vos références pour ce dossier *(facultatif)* BFF020343

1 NOM ET ADRESSE DU DEMANDEUR OU DU MANDATAIRE À QUI LA CORRESPONDANCE DOIT ÊTRE ADRESSÉE

CABINET PLASSERAUD
84, rue d'Amsterdam
75440 PARIS CEDEX 09

Confirmation d'un dépôt par télécopie ☐ N° attribué par l'INPI à la télécopie

2 NATURE DE LA DEMANDE — Cochez l'une des 4 cases suivantes

Demande de brevet	☒	
Demande de certificat d'utilité	☐	
Demande divisionnaire	☐	
Demande de brevet initiale	N°	Date
ou demande de certificat d'utilité initiale	N°	Date
Transformation d'une demande de brevet européen *Demande de brevet initiale*	☐ N°	Date

3 TITRE DE L'INVENTION (200 caractères ou espaces maximum)

PROCEDE ET DISPOSITIF D'ANALYSE DE LA SECURITE D'UN SYSTEME D'INFORMATION

4 DÉCLARATION DE PRIORITÉ OU REQUÊTE DU BÉNÉFICE DE LA DATE DE DÉPÔT D'UNE DEMANDE ANTÉRIEURE FRANÇAISE

Pays ou organisation
Date N°
Pays ou organisation
Date N°
Pays ou organisation
Date N°

☐ **S'il y a d'autres priorités, cochez la case et utilisez l'imprimé «Suite»**

5 DEMANDEUR (Cochez l'une des 2 cases) ☒ **Personne morale** ☐ **Personne physique**

Nom ou dénomination sociale		EADS DEFENCE AND SECURITY NETWORKS
Prénoms		
Forme juridique		Société par Actions Simplifiée
N° SIREN		
Code APE-NAF		
Domicile ou siège	Rue	Rue Jean-Pierre Timbaud - Batiment Jean-Pierre Timbaud
	Code postal et ville	7 8 1 8 0 MONTIGNY LE BRETONNEUX
	Pays	FRANCE
Nationalité		Française
N° de téléphone *(facultatif)*		N° de télécopie *(facultatif)*
Adresse électronique *(facultatif)*		

☐ **S'il y a plus d'un demandeur, cochez la case et utilisez l'imprimé «Suite»**

Remplir impérativement la 2ème page

INPI
INSTITUT NATIONAL DE LA PROPRIETE INDUSTRIELLE

BREVET D'INVENTION
CERTIFICAT D'UTILITE

REQUÊTE EN DÉLIVRANCE
page 2/2

BR2

Réservé à l'INPI

REMISE DES PIÈCES
DATE 14 NOV 2002
LIEU 75 INPI PARIS B
N° D'ENREGISTREMENT NATIONAL ATTRIBUÉ PAR L'INPI 0214279

DB 540 W / 210502

6 MANDATAIRE *(s'il y a lieu)*		
Nom		
Prénom		
Cabinet ou Société		CABINET PLASSERAUD
N °de pouvoir permanent et/ou de lien contractuel		
Adresse	Rue	84, rue d'Amsterdam
	Code postal et ville	7 5 0 0 9 PARIS
	Pays	
N° de téléphone *(facultatif)*		
N° de télécopie *(facultatif)*		
Adresse électronique *(facultatif)*		
7 INVENTEUR (S)		**Les inventeurs sont nécessairement des personnes physiques**
Les demandeurs et les inventeurs sont les mêmes personnes		☐ Oui ☒ Non : **Dans ce cas remplir le formulaire de Désignation d'inventeur(s)**
8 RAPPORT DE RECHERCHE		**Uniquement pour une demande de brevet (y compris division et transformation)**
Établissement immédiat ou établissement différé		☒ ☐
Paiement échelonné de la redevance *(en deux versements)*		**Uniquement pour les personnes physiques effectuant elles-mêmes leur propre dépôt** ☐ Oui ☐ Non
9 RÉDUCTION DU TAUX DES REDEVANCES		**Uniquement pour les personnes physiques** ☐ Requise pour la première fois pour cette invention *(joindre un avis de non-imposition)* ☐ Obtenue antérieurement à ce dépôt pour cette invention *(joindre une copie de la décision d'admission à l'assistance gratuite ou indiquer sa référence)* : **AG**
10 SÉQUENCES DE NUCLEOTIDES ET/OU D'ACIDES AMINÉS		☐ Cochez la case si la description contient une liste de séquences
Le support électronique de données est joint		☐
La déclaration de conformité de la liste de séquences sur support papier avec le support électronique de données est jointe		☐
Si vous avez utilisé l'imprimé «Suite», indiquez le nombre de pages jointes		

11 SIGNATURE DU DEMANDEUR OU DU MANDATAIRE
(Nom et qualité du signataire)
Stéphane VERDURE
CABINET PLASSERAUD
CPI n°97-0901

VISA DE LA PRÉFECTURE OU DE L'INPI

L. MARIELLO

La loi n°78-17 du 6 janvier 1978 relative à l'informatique, aux fichiers et aux libertés s'applique aux réponses faites à ce formulaire.
Elle garantit un droit d'accès et de rectification pour les données vous concernant auprès de l'INPI.

PROCEDE ET DISPOSITIF D'ANALYSE DE LA SECURITE D'UN SYSTEME D'INFORMATION

La présente invention se rapporte au domaine des outils d'analyse et de maîtrise de la sécurité des systèmes d'information (SSI).

Les outils de conception structurée d'un système d'information (tels que SADT ou SART) ne prennent pas en compte la sécurité du système. Les méthodes généralistes d'analyse du risque (telles que Marion, Melisa, ou CRAMM) sont peu précises, et incapables de mettre en évidence les failles techniques d'un système. Les outils de sûreté de fonctionnement (SDF) sont limités aux problèmes de fiabilité et de disponibilité, mais ne prennent pas en compte la malveillance. Les systèmes de détection d'intrusion (IDS pour "Intrusion Detection Systems") et les analyseurs de vulnérabilité ne fonctionnent que sur des systèmes réels, sont spécifiques d'un domaine restreint, et n'ont qu'une vision partielle de la sécurité. Les éditeurs de stratégie de sécurité fonctionnent selon le point de vue du défenseur seulement, et ne comportent pas de fonction d'analyse de la cohérence de la sécurité, ni de recherche des failles et des attaques possibles.

L'invention a pour objet un procédé et un dispositif d'analyse de la sécurité des systèmes d'information qui repose sur la modélisation et la simulation du système et des attaques possibles, pour analyser et maîtriser la sécurité du système.

Plus particulièrement, un premier aspect de l'invention concerne un procédé d'analyse de la sécurité d'un système d'information comprenant :

- une phase de modélisation comprenant la modélisation du système d'information ; et,
- une phase de simulation, comprenant la spécification et la simulation d'attaques potentielles contre le système d'information.

La phase de modélisation comprend la spécification de l'architecture du système avec un ensemble de composants du système et des relations entre lesdits composants, c'est-à-dire suivant un modèle composants / relations.

De préférence, des états déterminés sont associés à chaque composant du système d'information, chaque état pouvant prendre une valeur

saine et une ou plusieurs valeurs non saines. Certains au moins desdits états se rapportent respectivement à l'activité, à la confidentialité, à l'intégrité et/ou à la disponibilité du composant auquel ils sont associés.

De préférence, la phase de modélisation comprend en outre la spécification d'un ensemble de règles de comportement, notamment au plan du fonctionnement du système et au plan de la sécurité (règles de protection), associées aux composants du système.

Selon un avantage de l'invention, l'utilisateur peut adopter le point de vue du défenseur pendant la phase de modélisation, et le point de vue de l'attaquant pendant la phase de simulation. En itérant des phases de modélisation et des phases de simulation alternées, il parvient à maîtriser la sécurité du système d'information.

Selon un autre avantage, l'invention permet d'analyser la sécurité d'un système d'information sans intervention directe sur celui-ci. En effet, le lancement des attaques se fait sur un système virtuel, correspondant au système réel modélisé.

Avantageusement, l'analyse de la sécurité peut avoir lieu très tôt dans le processus de conception du système d'information, et notamment avant son déploiement physique.

Le procédé permet de modéliser un système d'information, en se limitant à ses aspects sécurité, et donc en restant maîtrisable par une personne humaine, grâce à des concepts et principes simplificateurs. L'invention permet de traiter d'une façon générique les aspects notamment techniques (c'est-à-dire liés aux caractéristiques techniques du système d'information), humains, procéduraux et physiques (par exemple géographiques). Un bon compromis entre réalisme et simplicité de la modélisation permet à un utilisateur (typiquement le concepteur ou l'administrateur du système d'information, ou un auditeur technique de la sécurité) de modéliser le système d'information sans avoir à le reproduire complètement.

Le procédé permet aussi de simuler avec réalisme toutes les attaques connues dans le monde des systèmes d'information. Ces attaques s'avèrent génériques, et peuvent s'appliquer aux domaines connexes des systèmes d'information (sécurité physique par exemple).

Il permet enfin de simuler avec réalisme toutes les parades connues dans le monde des systèmes d'information, qu'elles soient de prévention ou de détection. La modélisation des parades de prévention est réalisée par des règles de sécurité. La modélisation des parades de détection est réalisée au moyen de métriques.

En résumé, le procédé permet d'analyser en peu de temps la faisabilité d'un très grand nombre d'attaques sur un système donné. Avec un peu d'expérience, l'utilisateur peut simuler environ 100 à 200 attaques élémentaires par jour, ce qui est considérablement plus que ce qu'on pourrait faire sur un système réel.

Un second aspect de l'invention concerne un dispositif pour la mise en œuvre d'un procédé selon le premier aspect. A cet effet, le dispositif comprend avantageusement une interface Homme / machine pour la mise en œuvre de la phase de modélisation et/ou un moteur attaques / parades pour la mise en œuvre de la phase de simulation

Avantageusement, l'interface Homme / machine présente une fonctionnalité d'affichage en multi vues du système modélisé.

De préférence, l'interface Homme / machine permet d'afficher le système modélisé selon un modèle composants / relations.

Le dispositif est destiné à être utilisé en tant qu'outil d'audit technique de la sécurité des systèmes d'information existants (c'est-à-dire déjà déployés), pour lesquels il présente l'avantage de ne pas les perturber pendant leur exploitation. Il peut notamment analyser des attaques destructrices sans affecter le système réel.

Le dispositif peut aussi être utilisé en tant qu'outil d'aide à la conception de la sécurité de systèmes en cours de conception ou de réalisation, ce qui permet la mise au point et le test de leur politique de protection avant même leur installation.

Il peut aussi être utilisé en tant qu'outil privilégié de certification de système, au sens de la certification sécurité selon la normalisation existante: TCSEC, ITSEC ("Information Technology Security Evaluation Criteria") et CC ("Common Criteria"). Cette normalisation permet en effet actuellement de certifier des produits aux contours nettement délimités, mais est jusqu'à

maintenant inapplicable aux systèmes. Ces derniers sont en effet trop vastes, complexes, hétérogènes, aux contours mal définis et évolutifs, pour être évalués avec les techniques d'évaluation de produits.

En résumé, le dispositif est un outil destiné à des spécialistes de la sécurité: administrateurs système, auditeurs techniques de la sécurité. A ces personnes, il apporte en plus la possibilité d'adopter le point de vue de l'attaquant, point de vue absolument nécessaire pour construire une protection efficace et adaptée.

D'autres caractéristiques et avantages de l'invention apparaîtront encore à la lecture de la description qui va suivre. Celle-ci est purement illustrative et doit être lue en regard des dessins annexés sur lesquels :

- la figure 1 est un diagramme illustrant les phases de modélisation et de simulation selon le procédé de l'invention ;
- la figure 2 est un schéma synoptique illustrant les éléments d'un dispositif selon l'invention ;
- la figure 3 est un schéma illustrant la construction d'un modèle composants / relations pour un système d'information ;
- la figure 4 est un schéma montrant un exemple de modèle composants / relations, sur lequel apparaissent des relations d'hébergement et des relations d'échange entre des composants ;
- la figure 5 est un schéma illustrant des exemples de relations de service entre des composants d'un système d'information modélisé selon l'invention ;
- la figure 6 est un schéma qui illustre un exemple détaillé de la phase de simulation du procédé selon l'invention
- les figures 7a à 7^{e} sont des schémas qui illustrent des exemples de chemins d'attaque ;
- la figure 8 est un schéma montrant la transmission d'une attaque à travers un chemin d'attaque dans un système d'information modélisé ;
- la figure 9 et la figure 10 sont des schémas présentant un mécanisme de piles amont et aval ;
- la figure 11 est un schéma illustrant un exemple de fonctionnement des piles amont et aval ;

- la figure 12 est un diagramme illustrant l'algorithme général de fonctionnement du moteur attaques / parades ;

- la figure 13 est un diagramme qui illustre un mécanisme d'application des règles de protection ;

- les figures 14a et 14b sont des diagrammes illustrant un mécanisme de routage fonctionnel et un mécanisme de contagion d'états, respectivement ;

- la figure 15 est un schéma illustrant des techniques de détournement dans un système d'information modélisé ;

- la figure 16 est un schéma illustrant l'affichage d'un système modélisé, avec une fonctionnalité multi-vues ;

- la figure 17 est schéma illustrant l'affichage multi vues selon une approche hiérarchique ; et,

- la figure 18 est un schéma synoptique d'un dispositif selon l'invention.

Comme illustré par le graphe de la figure 1, le procédé comporte deux phases d'utilisation, elles même décomposées en deux étapes.

Pendant une phase de modélisation, l'utilisateur adopte le point de vue du défenseur. La phase de modélisation comprend une étape 1 de spécification de l'architecture du système avec un ensemble de composants du système et des relations entre lesdits composants, qui comprennent des relations de propagation et des relations de service. L'étape 1 aboutit à l'architecture modélisée du système réel, appelée modèle composants / relations dans la suite, ce modèle étant sauvegardé sous la forme d'un fichier dans une mémoire 11. Ce modèle peut être représenté de façon graphique par un graphe comportant des boîtes symbolisant les composants, reliées par des flèches symbolisant les relations. La phase de modélisation comprend aussi une étape 2 de spécification de règles de comportement, à ne pas confondre avec les relations précitées. Ces règles de comportement définissent le fonctionnement de chaque composant, aux plans du fonctionnement et de la sécurité, en particulier les protections qu'il assure. L'étape 2 aboutit à la construction d'un fichier de règles de comportement, qui est sauvegardé dans une mémoire 12.

A l'inverse, pendant une phase de simulation, l'utilisateur adopte le point de vue de l'attaquant. La phase de simulation comprend une étape 3 de

spécification des attaques, qui aboutit à la création d'un ou plusieurs scénarios d'attaques. Ces scénarios sont sauvegardés sous la forme de fichiers dans une mémoire 13. La phase de simulation comprend aussi une étape 4 de simulation interactive d'attaques. Cette simulation est réalisée par un moteur attaques / parades. Elle aboutit à la création d'un journal des attaques, qui est sauvegardé sous la forme d'un fichier dans une mémoire 14.

Les fichiers sauvegardés dans les mémoires 11, 12, 13 et 14 peuvent être importés de ou exportés vers différentes applications, en sorte que leur réutilisation est possible.

Les phases de modélisation 10 et de simulation 20 peuvent être itérées de façon alternée, en modifiant à chaque fois tout ou partie des paramètres (modèle composants / relations, règles de comportement, attaques) pris en compte. Ceci permet une bonne maîtrise de la sécurité du système par l'utilisateur.

Dans la suite de la description, il est décrit un mode de mise en œuvre des étapes 1 à 4 du procédé, avec toutefois une présentation modifiée quant à l'ordre des étapes. En effet, l'étape 3 (spécification d'attaques) et l'étape 4 (simulation interactive par moteur attaques / parades) sont exposées avant l'étape 2 (paramétrage des règles de protection), pour aider à la compréhension du texte. En outre, la description de ces étapes est complétée par une présentation de deux mécanismes complémentaires : les métriques et l'interface Homme / machine (IMM).

Au préalable, les principaux éléments d'un dispositif selon l'invention sont présentés en référence au diagramme fonctionnel de la figure 2. Sur cette figure, on a représenté en partie haute un groupe 20a des éléments utilisés pendant la phase de modélisation, et en partie basse un groupe 20b de ceux utilisés pendant la phase de simulation. De plus, une interface Homme / machine 15 est représentée sur la droite.

Le groupe 20a comprend un langage de règles 21 permettant d'élaborer les règles de protection 22, un ensemble de métriques de base 23, et le modèle composants / relations 24.

Le groupe 20b comprend un langage d'attaques 25 permettant d'élaborer les scénarios d'attaques 26, le moteur attaques / parades 16, un

mécanisme de piles amont et aval 27, un ensemble d'états potentiels 28 et des métriques calculées 29.

A la figure 2, les flèches en traits continus symbolisent les transferts d'informations entre les éléments, et les traits discontinus symbolisent les liens fonctionnels entre les éléments. Le rôle de chacun de ces éléments apparaîtra dans ce qui suit.

La première étape du procédé, étape 1, consiste à construire le modèle composants / relations (ou architecture) du système réel via l'interface humaine 15. Pour cela, le système est décomposé en composants élémentaires (ou atomes), dotés d'attributs, et reliés par des relations.

Les étapes de la construction du modèle composants / relations sont illustrés par le graphe de la figure 3.

Une première étape 31, l'utilisateur met en place sur un graphe à deux dimensions les composants qui forment le système. Dans un exemple, chaque composant est représenté par un rectangle contenant ses quatre états potentiels et son nom.

Les composants peuvent représenter toutes sortes d'objets réels hétérogènes de type physique (site, bâtiment, étage, local, bureaux coffre, porte, fenêtre, serrure, clé,...), support (papier, disque fixe, disquette, CDROM, bande magnétique,...), réseau (électrique, informatique, téléphonique, hertzien, aérien,...), matériel (mécanique, réseau, informatique, poste, serveur, écran, clavier, imprimante,...), logiciel (OS, driver, application, service,...), données (répertoire, fichier, information, base de données, mot de passe,...), personnes (utilisateur, agresseur, directeur, administrateur, organisation, service, ...), etc. Cette liste indicative et non limitative.

Chaque composant a plusieurs attributs, qui sont définis dans une étape 32. Les attributs comprennent des attributs statiques et des attributs dynamiques. Les attributs statiques comprennent par exemple un nom (composé d'une nature et d'un identifiant), et éventuellement un ou plusieurs adjectifs. Les attributs dynamiques comprennent par exemple des états potentiels dits états "ACID" (voir plus loin), un nom prétendu (dans le cas où le composant est usurpateur), et un lien vers un composant usurpateur (dans le cas où le composant est usurpé).

Les adjectifs ont pour but de permettre de désigner des composants sans les nommer, et ceci soit dans des règles (par exemple: tous les composants "externes" sont interdits d'accès), soit dans des attaques (par exemple: découvrir tous les composants "IP"). La liste des adjectifs est ouverte, et définie par l'utilisateur lors de la modélisation. Un exemple d'adjectifs classés par utilité est donné par la tableau I ci-dessous :

utilité	adjectifs
appartenance	interne, externe, privé, public, établissement, groupe, central, local, distant
sensibilité, gravité	normal, important, critique, vital, droits réservés (DR), confidentiel défense (CD), secret défense (SD), top-secret défense (TSD), tactique, stratégique
droit d'accès, rôle technique ou organisationnel, interne ou externe	usager, rédacteur, lecteur, éditeur, technicien, opérateur, exploitant, administrateur, stagiaire, employé, gardien, surveillant, responsable, gestionnaire, directeur, client, partenaire, consultant, fournisseur, client, concurrent
confiance ou méfiance	fiable, fidèle, sincère, sérieux, connu, inconnu, douteux, suspect, pirate
technique	IP, réseau, informatique, OS, exécutable, information, fixe, mobile.
protection	crypté, caché, secouru, répertorié, administré, dupliqué

Tableau I

La modélisation étant une tâche complexe pour l'utilisateur. C'est pourquoi les mécanismes de modélisation sont conçus pour fonctionner même sur un modèle non complètement terminé, donc éventuellement partiellement incohérent. Ceci permet à l'utilisateur de travailler selon un processus progressif et itératif, alternant la modélisation et les tests de fonctionnement par simulation.

Les composants sont ubiquistes et intemporels. Ce principe a pour but de simplifier le travail de modélisation, sans nuire au réalisme, dans la mesure où le thème est la sécurité. Il se traduit concrètement par le fait qu'un composant peut être, notamment :

- multiple, c'est-à-dire représenter plusieurs objets réels semblables situés en plusieurs endroits ;

- dispersé, c'est-à-dire représenter un objet de grande taille non situé dans une zone précise (par exemple un réseau);

- partiel, c'est-à-dire représenter une partie d'un objet réel complexe ;

- répliqué, lorsque deux composants peuvent représenter le même objet réel (par commodité de modélisation) ;

- dupliqué, lorsque, par exemple, un même fichier réel présente plusieurs copies sur plusieurs supports ;

- temporaire, c'est-à-dire représenter par exemple une communication téléphonique intermittente ; et

- mobile : par exemple, un poste portable, ou une personne sont mobiles ("nomades") par nature.

Dans une étape 33, ont définit des relations de propagation associées aux composants. Ces relations sont bidirectionnelles, et susceptibles de véhiculer des attaques dans les deux sens. Elles peuvent être de deux types distincts : de type hébergement (contenance, alimentation) et de type échange.

Le schéma de la figure 4 illustre un exemple de relations d'hébergement (symbolisées par des flèches verticales) entre un logiciel client 41, un poste de travail 42 (un ordinateur personnel ou PC) dans lequel ledit logiciel client est sauvegardé, et un bureau 43 dans lequel ledit poste de travail est situé. Le même schéma illustre aussi des relations d'échange entre le logiciel client 41 et un logiciel serveur 44 avec lequel le logiciel client échange des données, entre le poste de travail 42 et un réseau informatique auquel le poste de travail est raccordé 45, et entre le bureau 43 et un couloir 46 permettant l'accès audit bureau.

Quand un composant est traversé par une attaque, il peut ou non laisser une trace du passage dans l'attaque. Par exemple, un paquet postal porte le cachet du bureau de poste émetteur, mais pas du bureau de poste récepteur. De même, un paquet IP ("Internet Protocol") contient ou non l'adresse IP d'un routeur traversé.

Pour prendre en compte cette réalité, chaque composant, pour chacune des relations qu'il reçoit, a un indicateur de transparence ou opacité. S'il est transparent, il ne sera pas vu des composants avals sur le chemin de l'attaque, et ces composants ne pourront donc pas en tenir compte dans leurs

règles de protection. Si au contraire il est opaque, les composants avals le verront, mais il cachera les composants amonts de même type (c'est à dire ayant les mêmes adjectifs).

A côté des relations de propagation, on prévoit aussi des relations de service, qui sont définies dans une étape 34. Les relations de service sont unidirectionnelles, et ne véhiculent pas d'attaques. Elles ont pour but de permettre de désigner un composant à partir d'un autre, sous la forme d'une indexation.

Le schéma de la figure 5 illustre un exemple de relation de service. Sur cette figure, on a représenté un poste de travail 51, une personne 52, un mot de passe 53 et un bureau 54. Des relations de propagation entre ces composants sont symbolisées par des traits continus, et des relations de service sont symbolisées par des traits discontinus le composant. Dans cet exemple, le composant "mot de passe" 53 peut être désigné par la formule "passe(personne)". De même, le composant "bureau B24" peut être désigné par "bureau(personne)".

De retour à la figure 3, on notera que les étapes 31 à 34 peuvent être itérées pour permettre à l'utilisateur d'affiner la construction du modèle composants / relations.

Quand le modèle composants / relations est terminé, une table de routage est construite dans une étape 35. Cette table est calculée automatiquement selon le principe du plus court chemin entre les composants, en utilisant les relations de propagation uniquement. Le résultat est par exemple une matrice composant de départ / composant d'arrivée.

L'évolution dans le temps de chaque composant est mémorisée au moyen de quatre états potentiels, appelés états "ACID" (A = activité, C = confidentialité, I = intégrité, D = disponibilité). Ces états correspondent aux trois domaines connus de la sécurité (C, I, D), auxquels a été ajouté l'état "A" pour "activité". Ce dernier état représente la capacité d'un composant à agir, soit de façon normale (licite), soit de façon malveillante, et ceci sous l'influence de l'agresseur.

On pourra noter que ces états ACID correspondent sensiblement aux droits élémentaires des modèles de sécurité: "XRWD" (X = execute, R = read, W = write, D = delete). Ils sont indépendants les uns des autres.

Dans un exemple chaque état a quatre valeurs possibles : normal, affaibli, dégradé, dangereux. Dans un exemple, ces valeurs sont codées sur le graphe composants / relations par un code de couleur (par exemple vert, jaune, rouge, bleu foncé, respectivement), utilisé pour le remplissage du rectangle symbolisant chaque composant.

Le tableau II ci-dessous donne la signification des quatre valeurs possibles des quatre états ACID potentiels.

Etats ACID :	Activité	Confidentialité	Intégrité	Disponibilité
sain	fermé	discret	intègre	disponible
affaibli	ouvert	découvert	usurpateur	saturé
dégradé	actif	divulgué	altéré	bloqué
dangereux	interactif	trahi	corrompu	détruit

Tableau II

Le but de l'analyse est d'étudier la possibilité de réaliser une intrusion dans un système. Dans la mesure où il est nécessaire de faire des simplifications par rapport à la réalité, il est préférable de simplifier dans le sens du pessimisme, ceci dans un but sécuritaire. En d'autres termes, le dispositif peut voir des intrusions là où il n'y en n'a pas, mais ne doit pas oublier des intrusions réellement possibles. L'utilisateur fera ensuite la part des choses, si nécessaire.

Ce principe, dit de la "politique du pire" est appliqué dans le moteur attaques / parades de la manière suivante.

Au départ de la modélisation, les quatre états potentiels de chaque composant sont respectivement initialisés à la valeur "sain". L'utilisateur peut les initialiser manuellement à une valeur différente s'il le souhaite. Les états ne peuvent ensuite évoluer que vers la dégradation, au fur et à mesure des attaques réussies.

Les états sont "potentiels", c'est-à-dire qu'ils signifient que les composants "peuvent" se trouver dans les états indiqués, mais pas forcément.

Ceci implique que deux états apparemment incompatibles (par exemple, un composant à la fois "actif" et "détruit") peuvent coexister. Quand deux états sont incompatibles ou incohérents, le dispositif choisit la situation la plus défavorable au défenseur.

Quand un composant (modélisé) représente plusieurs objets réels, ses états potentiels sont ceux de l'objet réel le plus dégradé.

Enfin, le test d'un état de composant dans une règle ne correspond pas à une relation d'égalité, mais à une relation d'ordre. Par exemple, le test "composant = bloqué" est positif si le composant est soit "bloqué", soit "détruit".

On va maintenant décrire un exemple de mise en œuvre de l'étape 3 et de l'étape 4 du procédé, à savoir la spécification des scénarios d'attaque et leur simulation interactive. A cet effet, il est fait référence au diagramme de la figure 6.

Dans une étape 61, une attaque est définie. Dans un exemple, il est prévu une attaque pour chaque dégradation d'un état ACID. Les attaques correspondantes sont appelées attaques élémentaires dans la suite. Elles sont liées directement aux états ACID. Par exemple, pour faire passer un composant à l'état "bloqué", on utilise l'attaque "bloquer".

Il y a donc douze attaques élémentaires (correspondant aux douze valeurs d'états potentiels non saines), qui sont résumées dans le tableau II ci-dessous, plus une attaque spéciale dite attaque d'usurpation. Cette dernière attaque, appelée "ChangerNomPrétendu", permet à un composant d'usurper le nom d'un autre. L'usurpation est en effet une technique fondamentale de l'intrusion.

attaques ACID	Activité	Confidentialité	Intégrité	Disponibilité
faible	ouvrir	découvrir	usurper	saturer
grave	activer	espionner	altérer	bloquer
dangereuse	pénétrer	trahir	corrompre	détruire

Tableau III

Lors de la définition d'une attaque élémentaire, l'utilisateur (qui se place du point de vue de l'attaquant) saisit les paramètres suivants :

- type d'attaque, parmi les 13 ci dessus ;
- type de protocole (voir ci dessous) ; et,
- éléments du chemin (voir plus loin).

Le "protocole" généralise le concept de protocole de télécommunication, et désigne tout moyen de transmission d'une attaque entre deux composants. Une attaque se concrétise toujours par la transmission dans le système d'un objet réel, physique ou logique, véhiculant des informations, des logiciels ou des mécanismes malveillants.

La liste des protocoles est ouverte, l'utilisateur pouvant en définir autant qu'il souhaite lors de la modélisation. Des exemples de protocoles sont :

- protocoles physiques : lettre, colis, disquette, CDROM, personne,...
- protocoles logiques et informatiques : mail, web, fichier, Telnet, Netbios, TCP/IP, FTP, SSL,...
- protocoles spécialisés : missile, onde hertzienne, laser, ...
- etc.

Le langage d'attaque utilise les mêmes mots que le langage de règles, et permet de définir des scénarios d'attaques complexes. Une ligne d'attaque élémentaire est par exemple :

"départ=agresseur + intermédiaire=Internet + protocole=mail + arrivée=usager + attaque=détruire + cible=poste(usager)"

En français, cette ligne d'attaque signifie que l'agresseur envoie, via Internet, un courrier électronique à l'usager, contenant une bombe logique qui va détruire son poste.

Le chemin d'attaque est l'un des concepts centraux du dispositif, dont l'objet est justement de trouver des chemins d'attaque dans un système modélisé. Lors de la simulation, l'utilisateur spécifie le chemin sous la forme suivante :

- un composant de départ ;
- un composant d'arrivée ;
- un composant cible ; et, éventuellement
- un composant intermédiaire.

L'agresseur et la cible doivent nécessairement se trouver sur le chemin, mais pas nécessairement au début ou à la fin. Le schéma des figures

7a à 7e montrent diverses possibilités (non limitatives) de chemin d'attaque, qui peuvent correspondre chacune à un cas réel. Par exemple, lorsque la cible est un poste de travail :

- à la figure 7a, l'agresseur sature le poste de travail par du trafic artificiel transmis via le réseau ;

- à la figure 7b, l'agresseur envoie un virus par courrier électronique qui est exécuté par un usager "pigeon", ce qui altère le poste de travail ;

- à la figure 7c, l'agresseur est un serveur pirate, et c'est un usager "pigeon qui consulte une page web sur serveur pirate et reçoit un applet Java pirate qui altère le poste ;

- à la figure 7d, l'agresseur est aussi un serveur pirate et le poste de travail (rendu actif) ouvre une "reverse back door" vers le serveur pirate ; et,

- à la figure 7e, l'agresseur est un réseau altéré par lequel passe une session, dévoilant ainsi des informations du poste de travail.

Bien entendu, les exemples ci-dessus sont illustratifs et nullement limitatifs.

De retour à la figure 6, l'attaque spécifiée à l'étape 61 est lancée dans une étape 62. Dans une étape 63, elle est exécutée par le moteur d'attaques / parades. Et dans une dernière étape 64, le résultat de l'attaque est constaté. Les étapes 61 à 64 sont itérées, étant exécutées pour chaque attaque du scénario d'attaques.

Le résultat d'une attaque, si elle réussit, est de faire passer l'un des états ACID du composant cible à une valeur dégradée (non saine), voire à une valeur dégradée plus grave si il était déjà une valeur dégradée. Par exemple dans le cas de l'attaque "bloquer" :

- si le composant est moins dégradé que "bloqué" (par exemple, "saturé"), alors il devient "bloqué" ;

- s'il est déjà "bloqué", alors il reste "bloqué" ; et,

- s'il est plus dégradé que "bloqué" (par exemple "détruit"), alors il ne change pas d'état.

On va maintenant décrire plus en détails le fonctionnement du moteur d'attaques / parades. Typiquement, la transmission d'une attaque sur le chemin

d'attaque se fait selon trois phases successives, illustrées par le schéma de la figure 8, à savoir :

- une phase de propagation durant laquelle l'attaque traverse les composants vecteurs, qui peuvent ou non assurer un filtrage de sécurité via leurs règles de protection ;

- une phase d'absorption durant laquelle l'attaque peut ou non dégrader la cible, selon ses défenses propres (règles de protection) ; et,

- une phase de contagion durant laquelle la dégradation de la cible peut se communiquer à d'autres composants sans qu'ils puissent se défendre (par exemple, l'explosion d'un bureau entraîne la destruction des équipements présents dans ce bureau).

Quand une attaque arrive dans un composant, le moteur attaques / parades effectue pour lui un certain nombre de contrôles (règles), basés sur les états de certains composants, et en particulier sur les composants situés en amont et en aval sur le chemin de l'attaque. Pour ces derniers, représentés sur le schéma de la figure 9, les informations dont dispose le moteur attaques / parades sont consignées respectivement dans la pile amont et dans la pile aval qui ont été présentées plus haut en regard du schéma de la figure 2. Ces piles amont et aval sont la modélisation des indications portées sur les objets réels (par exemple adresses et cachets de la Poste sur un colis, adresses IP sur un paquet IP, etc.).

La pile amont donne la liste (dans l'ordre) de tous les composants déjà traversés par l'attaque. Dans un exemple, il y a précisément deux piles amont : l'une qui contient la liste exhaustive de tous les composants, avec leur nom réel, et l'autre qui contient uniquement la liste des composants opaques, avec leur nom prétendu. La première liste est utilisée pour exécuter les règles de possibilité et de contagion du composant. L'autre est utilisée pour exécuter pour les règles de protection (identification, autorisation et routage).

La pile aval donne la liste des destinataires identifiés de l'attaque. Il y a un destinataire final, et éventuellement un ou plusieurs destinataires intermédiaires. Ces destinataires intermédiaires sont spécifiés, soit par l'utilisateur lors de la définition d'une attaque, soit par le routage fonctionnel. La pile aval est utilisée par les règles des composants.

Dans l'exemple illustré à la figure 10, le routeur 111 et l'internet 112 sont transparents (car l'adresse IP source des paquets IP venant de l'extérieur est inchangée). C'est pourquoi, étant empilés dans la première pile amont 110 des noms réels, et ils ne sont pas empilés dans la seconde pile amont 120 des noms visibles (c'est-à-dire réel ou prétendu, le cas échéant). Par ailleurs, le pirate 113 usurpe le nom d'un usager 114. Son "nom visible" (empilé dans la seconde pile amont 120) est donc "usager" et non "pirate". La pile aval est désignée par la référence 130.

Pour chaque transit d'une attaque, le moteur effectue les trois opérations suivantes :

- premièrement, il détermine le prochain composant destinataire de l'attaque: c'est le premier composant de la pile aval,
- deuxièmement, il détermine, grâce à la matrice de routage local, quelle est la relation à emprunter pour aller vers ce composant ; et,
- troisièmement, le composant qui se trouve de l'autre côté de la relation devient le composant en cours.

L'attaque s'arrête quand la pile aval est vide (et non quand le composant d'arrivée est atteint, car d'autres composants peuvent être empilés en aval après l'arrivée).

Pour le composant en cours (atteint par l'attaque), le moteur effectue les opérations suivantes, illustrées par le schéma de la figure 11 :

- il extrait le composant en cours de la pile aval, s'il y est (étape 1) ;
- il teste les règles de propagation du composant, qui peuvent accepter ou refuser l'attaque ;
- si l'attaque est refusée, le moteur l'arrête, sinon il continue les actions suivantes,
- dans le cadre des règles de routage fonctionnel, il peut empiler un composant intermédiaire en aval (étape 2) ;
- il empile le composant en cours dans la pile amont (étape 3) ; et
- si l'attaque est acceptée, il la route vers le composant suivant.

L'algorithme général de fonctionnement du moteur d'attaques / parades est donnée par le diagramme de la figure 12.

Sur ce diagramme, le mot-clé "moi" désigne le composant en cours. Pour transporter ou absorber une attaque, le composant en cours doit être dans l'état "ouvert" (état A des états ACID). Pour absorber l'attaque, le composant cible doit se trouver dans la première pile amont, celle des noms réels. On notera que les règles d'identification et d'autorisation donnent toujours "oui" pour résultat si le composant est corrompu.

A la figure 12, les trois phases de la transmission de l'attaque, à savoir la propagation, l'absorption et la contagion de l'attaque, sont représentées séparément.

On va maintenant décrire l'étape 2 du procédé, qui est le paramétrage des règles de protection des composants.

Lorsque l'architecture d'un système est spécifiée (à la fin de l'étape 1), cette architecture étant représentée par le graphe composants / relations, il reste en effet à décrire le comportement des composants, aux plans du fonctionnement et de la sécurité.

A cet effet, l'étape 2 consiste à paramétrer le fonctionnement de chaque composant, et en particulier à décrire les protections qu'il assure. Ceci est fait au moyen de la saisie de règles, structurées selon un langage et une grammaire de règles.

Le langage de règles se caractérise par, a capacité au réalisme (pour décrire précisément le fonctionnement d'un composant réel), l'ergonomie pour l'usager (simplicité, souplesse, naturel, peu de signes spéciaux) et la généricité (c'est-à-dire le fait de permettre une réutilisation facile des règles d'un modèle à l'autre).

Au plan de l'édition, le langage est de préférence sous forme de texte ASCII ou XML, est lisible, modifiable et imprimable avec un traitement de texte classique (tels que Notepad, MS-Office), accepte des commentaires en langage naturel sous la forme: /* commentaire */, accepte les majuscules / minuscules et accents (mais qui ne sont pas discriminants), et utilise les caractères "blanc" et "ligne suivante" uniquement pour la présentation.

De préférence, les mots du langage apparaissent de la même façon et sous la même forme dans la représentation graphique du modèle composants / relations, dans la spécification des règles (langage de règles),

dans la spécification des attaques (langage d'attaques), et dans le journal des attaques.

Le langage est principalement un langage de prédicats (ou critères de condition logique), utilisant des opérateurs logiques (tels que ET, OU, NON), des mots-clés (tels que "attaque", "amont", "aval", "protocole", "moi"), et des noms de composants sous forme directe ou par adjectif, ou encore par relation de service.

Par exemple, le prédicat "amont(person)=admin + attaque=corrompre" signifie que l'attaque "corrompre" est autorisée à passer dans le composant s'il y a en amont une personne ayant rôle "admin" (administrateur).

Un exemple de mots-clés génériques, qui est illustratif seulement, est donné par le tableau IV ci-dessous. Par ailleurs, à ces mots clés génériques, il y a lieu d'ajouter les seize valeurs des états potentiels et les douze valeurs des attaques.

mot-clé générique	désigne	signification	utilisable par	
			règles	attaques
amont	composant	en amont (règles possibilité et contagion)	+	
amont	composant	en amont (autres règles) sous nom prétendu	+	
aval	composant	en aval	+	
moi	composant	en cours de traitement	+	
entrée	composant	dernier traversé en amont (= relation d'entrée)	+	
départ	composant	premier en amont	+	+
cible	composant	cible de l'attaque	+	+
arrivée	composant	point d'arrivée de l'attaque	+	+
attaque	attaque	type d'une attaque	+	+
protocole	protocole	type de protocole	+	+
intermédiaire	composant	imposé sur le chemin de l'attaque		+
estlicite	mode	indique si l'attaque est licite ou non		+
présent	état	présence d'un composant dans une pile	+	
absent	état	absence d'un composant dans une pile	+	
authentique	état	état d'un composant non usurpé	+	
usurpé	état	état d'un composant usurpé par un autre	+	
usurpateur	composant	composant qui en usurpe un autre	+	

Tableau IV

Le langage de description des composants permet d'écrire des règles, qui comportent chacune un nombre variable de prédicats (condition logique booléenne) et éventuellement d'actions. Pour chaque composant, il existe huit types de règles. Les règles ont deux caractéristiques indépendantes :

- elles peuvent être binaires (condition logique booléenne, donnant une valeur oui / non) ou fonctionnelles (condition logique impliquant une action de routage ou de contagion) ; et,

- elles peuvent être "de propagation" (dans les composants vecteurs des attaques) ou "d'absorption" (dans les composants cibles des attaques).

Dans un exemple, cinq règles de propagation et trois règles d'absorption sont données respectivement par le tableau V et par le tableau VI ci-dessous.

règle	**type**	**rôle**	**exemple**
possibilité	binaire	définit quelles attaques *peuvent* passer	un logiciel ne peut pas transporter un colis postal
identification amont	binaire	assure l'identification et l'authentification des composants amonts	une attaque sera acceptée si le mot de passe a été préalablement divulgué
identification aval	binaire	assure l'identification et l'authentification des composants avals	
autorisation	binaire	définit quelles attaques *sont autorisées* à passer	un dispositif pare-feu ("firewall") peut interdire les attaques de découverte par paquets "Ping"
routage	fonctionnelle	définit vers quel composant diriger les attaques	un routeur envoie les messages mail vers le serveur mail

Tableau V

règle	**type**	**rôle**	**exemple**
possibilité	binaire	définit quelles attaques peuvent passer	un bureau peut absorber une bombe physique, mais pas une bombe logique
identification amont	binaire	assure l'identification et l'authentification éventuelle des composants amont	une attaque sera acceptée si le mot de passe a été préalablement divulgué
contagion	fonctionnelle	définit quels états doivent être propagés vers quels composants	l'explosion d'un bureau entraîne la destruction des équipements hébergés

Tableau VI

Le règles sont appliquées de la façon suivante. Chaque fois qu'une attaque se présente devant un composant (en cours), le moteur attaques / parades déroule le mécanisme illustré par le diagramme de la figure 13.

Pour qu'une attaque soit propagée par un composant vecteur ou soit absorbée par un composant cible, chaque catégorie de règle binaire doit, soit être vide, soit comporter une règle avec résultat positif.

Si l'attaque est acceptée par un composant vecteur, celui ci applique les règles de routage fonctionnel selon le mécanisme illustré par le diagramme de la figure 14a. De même, si l'attaque est acceptée par le composant cible, celui ci applique les règles de contagion selon le mécanisme illustré par le diagramme de la figure 14b.

On va maintenant décrire les mécanismes de routage local et fonctionnel. Le routage a pour fonction de diriger les attaques d'un composant à l'autre, dans le but d'atteindre le point d'arrivée ou les points intermédiaires. Il existe deux sortes de routages.

Premièrement, le routage fonctionnel, défini par l'utilisateur via les règles de routage, et qui permet de définir un nouveau composant intermédiaire avant d'atteindre le point d'arrivée. Ce composant est inséré dans la pile aval. Par exemple, un routeur décide de router un courrier électronique venant de l'extérieur vers le serveur de messagerie.

Deuxièmement, le routage local, invisible de l'utilisateur, qui a pour effet de diriger l'attaque vers le prochain composant de la pile aval, selon le chemin le plus court. La table de routage local est utilisée à cet effet. On rappelle que cette table est construite automatiquement en fin de modélisation, selon le principe de recherche du plus court chemin. Dans un exemple, elle est structurée sous forme de matrice (composant de) départ / arrivée.

Le routage fonctionnel permet d'assurer à la fois le fonctionnement nominal du système, et certaines fonctions de sécurité. Par exemple, un routeur qui envoi les paquets IP vers un dispositif par(feu ("firewall") assure une fonction de sécurité. Cette fonction de sécurité peut être dégradée dans le cas du détournement, ainsi qu'il va maintenant être décrit.

La technique de détournement, fondamentale dans les scénarios d'attaque, peut être prise en compte par le dispositif. Cette technique consiste à modifier la façon dont est fait le routage fonctionnel. Il trois sortes de détournements, illustrées par le diagramme de la figure 15.

A la figure 15 on a représenté un exemple de chemin de propagation allant d'un client 151 à un serveur 155 à travers (dans cet ordre) un premier routeur 152, un filtre 153 et un second routeur 154.

Un premier détournement est un détournement par contournement, symbolisé par la flèche 156. Dans l'exemple représenté, le routeur 152 est altéré en sorte que son composant aval, le filtre 153, est supprimé.

Un deuxième détournement est un détournement par interception, symbolisé par les flèche 158a et 158b. Dans l'exemple représenté, le routeur 152 est altéré en sorte qu'un composant aval 157 est ajouté sur le chemin de propagation. Ce composant aval est un pirate (usurpant). Dans ce cas, le serveur 155 est dit usurpé.

Un troisième détournement est un détournement par usurpation, symbolisé par la flèche 150. Dans l'exemple représenté, le routeur 154 est altéré en sorte que son composant aval, le serveur 155, est remplacé par un autre composant aval 159. Cet autre composant aval est un pirate (usurpant); le serveur 155 est dit usurpé.

Un détournement peut être réalisé par tout composant ayant une fonction de routage, situé dans le chemin de l'attaque. Il est déclenché par la réunion de trois conditions :

- le composant de routage a un état "altéré" (indiquant que ses tables de routage sont altérées) ;
- le composant d'arrivée est "usurpé" (du moins pour l'usurpation et l'interception) ; et,
- le composant de routage a une (ou plusieurs) règle particulière qui prévoit le détournement.

On va maintenant présenter un premier mécanisme supplémentaire de l'invention, constitué par les métriques de faisabilité et de non détection.

Les métriques ont pour but de compléter le langage de règles, qui est principalement axé sur la protection par prévention. Elles permettent de relativiser le succès ou l'échec des attaques, en calculant un coefficient de faisabilité et de non détection, apportant ainsi la protection par détection.

Dans un exemple, il existe cinq métriques dont trois métriques de base, qui sont paramétrées lors de la modélisation, et deux métriques de probabilité de sinistre, qui sont calculées lors de la simulation.

Pour éviter de tomber dans le travers de la complexité, les métriques de base sont évaluées sur un nombre restreint de niveaux, par exemple quatre niveaux. Cette échelle de niveaux doit être comprise comme une échelle logarithmique, c'est à dire que chaque niveau implique un coefficient multiplicateur par rapport au niveau inférieur. Les quatre niveaux correspondent par exemple aux valeurs 0.1%, 1%, 10%, 100%.

Deux métriques de base relèvent du point de vue du défenseur. Il s'agit d'une métrique d'efficacité des parades (résistance), et d'une métrique d'efficacité de détection des attaques. Ces 2 métriques sont paramétrées par le l'utilisateur pendant la phase de modélisation, indépendamment pour chaque règle de protection dans chaque composant, selon une échelle de valeurs telle que faible, moyen, fort, absolu.

En outre, une métrique de base relève du point de vue de l'attaquant. Il s'agit d'une métrique de moyens de l'attaquant. Cette métrique comprend par exemple les aspects suivants: compétence, outils, argent, temps. Elle est paramétrée par l'utilisateur pendant la phase de simulation, de façon globale pour l'attaquant, tous moyens, toutes attaques et toutes cibles confondus. L'échelle de valeurs est par exemple : public, initié, spécialiste, expert.

Les métriques de probabilité de sinistre sont calculées par le moteur attaques / parades lors du passage dans chaque composant, puis consolidées par le moteur sur tout le chemin, puis sur tout le scénario. Il s'agit d'une métrique de probabilité de passage d'une attaque sur un composant, d'une part, et d'une métrique de probabilité de non détection d'une attaque sur un composant, d'autre part. De préférence, elles sont exprimées sur l'échelle à quatre niveaux des métriques de base, à laquelle on ajoute le niveau 0%, et elles sont calculées selon les formule suivantes :

- probabilité de passage = (moyens de l'attaquant) / (efficacité de la protection) ;
- probabilité de non détection = (moyens de l'attaquant) / (efficacité de la détection).

Le tableau VII ci-dessous donne le calcul des métriques calculées.

efficacité du défenseur	0.1%	faible	0.1%	1.0%	10.0%	100.0%
	1.0%	moyen	0.0%	0.1%	1.0%	10.0%
	10.0%	fort	0.0%	0.0%	0.1%	1.0%
	100.0%	absolu	0.0%	0.0%	0.0%	0.1%
moyens de l'attaquant ⇨			public 0.1%	initié 1.0%	spécialiste 10.0%	expert 100.0%

Tableau VII

On va maintenant décrire un autre mécanisme supplémentaire de l'invention, qui fait partie de l'interface Homme / machine.

Avantageusement, l'interface Homme / machine présente une fonctionnalité dite "multi vues". Ceci n'est pas en soi une originalité, car la plupart des logiciels utilisent ce genre de fonctionnalité. Ce qui est original, par contre, c'est l'utilisation des vues pour aider l'utilisateur à maîtriser des systèmes complexes, grâce à une association entre les vues (logicielles) et les sous-systèmes (conceptuels).

Le système des "vues" est un élément important de l'interface Homme / machine, qui permet la modélisation de systèmes complexes. Son principe est de décomposer le système en plusieurs vues, dont une seule est affichée à l'écran dans la fenêtre principale, l'utilisateur pouvant passer alternativement d'une vue à l'autre. Tout composant peut être placé dans une vue, dans une autre, ou dans plusieurs vues, au choix de l'utilisateur.

Le schéma de la figure 16 montre un exemple de représentation graphique d'un système (modélisé) selon trois vues superposées. Le serpention symbolise un chemin d'attaque passant par les trois vues.

Il n'y a pas de contraintes pour la définition des vues, et pour la répartition des composants dans les différentes vues d'un système. On peut par exemple mettre en place des vues associées aux différents métiers du système (géographique, informatique, organisationnel).

Avantageusement, moyennant certains principes simples, pris isolément ou en combinaison, les vues deviennent cependant un élément de structuration fonctionnelle des systèmes.

Selon un premier tel principe, chaque vue représente de préférence un sous-système relativement autonome et indépendant du reste du système.

Selon un deuxième tel principe, on fait en sorte, de préférence, qu'il n'existe pas de relations de propagation ni de services entre deux vues. Seuls les composants communs à deux vues assurent la fonction d'interconnexion entre les vues.

Selon un troisième tel principe, enfin, les règles des composants situés dans une vue ne doivent pas faire appel nommément à des composants situés dans une autre vue.

Plus généralement, il y a deux façons de concevoir les vues. Soit les vues sont considérées comme des sous-systèmes de même niveau interconnectés entre eux (par exemple, des sites interconnectés via Internet, le composant commun étant Internet). Ou bien l'une des vues est considérée comme une description globale du système, tandis que les autres représentent des détails de tel ou tel composant complexe. Cette approche est appelée approche hiérarchique et va être détaillée maintenant.

L'approche hiérarchique est très puissante, car elle permet d'assembler des composants détaillés et mis au point préalablement pour former par assemblage des systèmes très complexes et réalistes, tout en restant simple à visualiser.

Dans cette approche, illustrée sous la forme d'un exemple donné à la figure 17, une vue supérieure présente le système global, qui contient un ou plusieurs composants représentant chacun un sous-système. Chaque vue inférieure donne le détail d'un sous système. A la figure, le serpentin symbolise un chemin d'attaque passant par les deux vues.

Un composant A commun à la vue supérieure et à une vue inférieure, appelé "composant relais", représente le sous système vu du système global, et réciproquement. Le composant relais est l'unique interface entre les deux vues.

Le composant relais assure l'étanchéité entre les vues, tout en assurant la communication entre elles, selon un triple rôle.

Le composant relais assure tout d'abord un rôle de relais de routage. En effet, il assure le routage des attaques dans les deux sens entre les deux

vues. Il utilise pour cela tous les critères de routage disponibles dans le langage de règles.

Le composant relais assure ensuite un rôle de relais de service. En effet, il peut avoir des relations de service, partant ou aboutissant, dans les deux vues. Ceci permet de désigner un composant d'une vue à l'autre via l'indexation.

Le composant relais assure enfin un rôle de relais de contagion d'état. A cet effet, il assure pour la vue supérieure une vision synthétique de la vue inférieure, via une contagion des principaux états représentatifs de cette vue.

A la figure 18, sur laquelle les mêmes éléments qu'aux figures 1 et 2 portent les mêmes références, on a représenté le schéma synoptique d'un exemple de réalisation d'un dispositif selon l'invention. Ce dispositif convient pour la mise en œuvre du procédé selon l'invention.

Dans cet exemple, le dispositif est implémenté dans un ordinateur à usage général comprenant un microprocesseur 10. L'interface Homme / machine 15 et le moteur attaques / parades 16 sont mis en œuvre sous la forme de modules logiciels, sauvegardés dans une mémoire 17 et plus particulièrement dans une mémoire à lecture seule (ROM). Ils sont exécutés par le microprocesseur 10 lorsqu'ils sont chargés dans la mémoire vive de l'ordinateur.

Pour assurer l'entrée de données par l'utilisateur, le dispositif comprend un clavier 19b, et en général également une souris (non représentée) ou similaire. Pour l'affichage des données, notamment pour l'affichage de la représentation graphique du système modélisés sous la forme d'une on plusieurs vues, le dispositif comprend aussi un écran. Ces éléments sont ceux qui équipent l'ordinateur.

Enfin, le dispositif comprend une mémoire vive 18, en particulier une mémoire à accès aléatoire (RAM), dans laquelle les fichiers 11, 12, 13 et 14 peuvent être sauvegardés.

REVENDICATIONS

1. Procédé d'analyse de la sécurité d'un système d'information comprenant :

- une phase de modélisation (1,2), comprenant la modélisation du système d'information,
- une phase de simulation, comprenant la spécification (3) et la simulation (4) d'attaques potentielles contre le système d'information.

2. Procédé selon la revendication 1, suivant lequel la phase de modélisation comprend la spécification (1) de l'architecture du système avec un ensemble de composants du système et des relations entre lesdits composants.

3. Procédé selon la revendication 2, suivant lequel, un nom étant associé à chaque composant, un ou plusieurs adjectifs peuvent aussi être associés audit composant, lesquels adjectifs permettent de désigner ledit composant sans le nommer.

4. Procédé selon la revendication 2 ou la revendication 3, suivant lequel des états déterminés sont associés à chaque composant du système d'information, chaque état pouvant prendre une valeur saine et une ou plusieurs valeurs non saines.

5. Procédé selon la revendication 4, suivant lequel certains au moins desdits états se rapportent respectivement à l'activité, à la confidentialité, à l'intégrité et/ou à la disponibilité du composant auquel ils sont associés.

6. Procédé selon l'une quelconque des revendications 2 à 5, suivant lequel un nom prétendu peut être associé à un composant déterminé quelconque, notamment dans le cas où ledit composant déterminé est usurpateur.

7. Procédé selon l'une quelconque des revendications 2 à 6, suivant lequel un lien vers un autre composant peut être associé à un composant déterminé quelconque, notamment dans le cas où ledit composant déterminé est usurpé et où ledit autre composant est usurpateur.

8. Procédé selon l'une quelconque des revendications 2 à 7, suivant lequel les relations entre deux composants déterminés quelconques, comprennent des relations de propagation bidirectionnelles susceptibles de véhiculer des attaques dans les deux sens.

9. Procédé selon l'une quelconque des revendications 2 à 8, suivant lequel les relations entre deux composants déterminés quelconques, comprennent des relations de service permettant de désigner un composant à partir d'un autre composant.

10. Procédé selon la revendication 2, suivant lequel la phase de modélisation comprend en outre la spécification (2) d'un ensemble de règles de comportement associées aux composants du système.

11. Procédé selon la revendication 10, suivant lequel chaque règle de comportement comprend un ou plusieurs prédicats, et/ou une ou plusieurs actions.

12. Procédé selon la revendication 10 ou la revendication 11, suivant lequel les règles de comportement comprennent des règles de propagation d'attaques, ces règles étant par exemple mises en œuvre dans des composants qui sont des vecteurs d'attaques, et des règles d'absorption d'attaques, ces règles étant par exemple mise en œuvre dans des composants qui sont la cible d'attaques.

13. Procédé selon l'une quelconque des revendications 10 à 12, suivant lequel les règles de comportement comprennent des règles binaires, par exemple des conditions logiques booléennes donnant une valeur de type oui / non, et/ou des règles fonctionnelles, par exemple des conditions logiques impliquant une action de routage (pour une règle de propagation) ou de contagion (pour une règle d'absorption).

14. Procédé selon l'une quelconque des revendications 2 à 13 comprenant, à la fin de la phase de modélisation (figure 3), la construction (35) d'une table de routage local, permettant de diriger une attaque d'un composant de départ vers un composant d'arrivée.

15. Procédé selon la revendication 14, suivant lequel la table de routage local est générée de façon automatique suivant le principe du plus court chemin entre le composant de départ et le composant d'arrivée.

16. Procédé selon l'une quelconque des revendications 3 à 15, suivant lequel l'étape de simulation des attaques comprend la mise à jour de l'état d'un composant du système altéré par une attaque réussie.

17. Procédé selon la revendication 116, suivant lequel la phase de simulation comprend en outre la constitution d'un fichier ou journal des attaques, contenant l'historique des changements de l'état des composants consécutifs à des attaques réussies, notamment pour permettre un traitement ultérieur par un utilisateur.

18. Procédé selon l'une quelconque des revendications précédentes, suivant lequel les attaques comprennent des attaques élémentaires correspondant à des valeurs d'états non saines.

19. Procédé selon l'une quelconque des revendications précédentes, suivant lequel les attaques comprennent en outre une attaque spéciale d'usurpation.

20. Procédé selon l'une quelconque des revendications précédentes, suivant lequel une attaque est définie, notamment, par un type d'attaque, un type de protocole, et des éléments de chemin d'attaque.

21. Procédé selon la revendication 20, suivant lequel les éléments de chemin d'attaque comprennent un composant de départ, un composant d'arrivée, un composant cible, et le cas échéant un ou plusieurs composants intermédiaires.

22. Procédé selon la revendication 20 ou la revendication 21, suivant lequel la liste des composants déjà traversés par une attaque est sauvegardée dans au moins une ou plusieurs piles amont.

23. Procédé selon la revendication 22, suivant lequel les pile amont comprennent une pile (110) contenant la liste exhaustive de tous les composants traversés, désignés par leur nom réel.

24. Procédé selon la revendication 22 ou la revendication 23, suivant lequel les piles amont comprennent une pile (120) contenant la liste des seuls composants traversés qui sont opaques, désignés par leur nom réel ou, le cas échéant, par leur nom prétendu.

25. Procédé selon l'une quelconque des revendications 20 à 24, suivant lequel la liste des composants destinataires d'une attaque est sauvegardée dans au moins une pile aval (130).

26. Procédé selon l'une quelconque des revendications 10 à 25, suivant lequel les attaques sont définies dans un langage utilisant les mêmes mots qu'un langage dans lequel les règles de comportement sont définies.

27. Procédé selon l'une quelconque des revendications précédentes, suivant lequel la phase de modélisation et/ou la phase de simulation sont mises en œuvre par un utilisateur au moyen d'une interface Homme/machine comportant une fonctionnalité multi vues, suivant laquelle une représentation graphique du système est présentée à l'utilisateur en plusieurs vues.

28. Procédé selon la revendication 27, suivant lequel chaque vue représente un sous-système du système, qui est relativement autonome et indépendant du reste du système.

29. Procédé selon la revendication 27 ou la revendication 28, suivant lequel la fonction d'interconnexion entre les composants compris dans deux vues distinctes est assurée seulement via le composant commun ou les composants communs aux deux vues.

30. Procédé selon l'une quelconque des revendications 27 à 29, suivant lequel les règles de comportement des composants appartenant à une vue ne font pas appel nommément à des composants appartenant à une autre vue.

31 Procédé selon l'une quelconque des revendications 27 à 30, suivant lequel les vues sont associées à des sous-systèmes respectifs, par exemple de même niveau, qui sont interconnectés entre eux via au moins un composant commun.

32. Procédé selon l'une quelconque des revendications 27 à 30, suivant lequel une vue supérieure est associée au système dans son ensemble, tandis qu'une ou plusieurs vues inférieures sont respectivement associées à un sous-système déterminé du système.

33. Procédé selon la revendication 32, suivant lequel un composant déterminé, commun à la vue supérieure et à une vue inférieure déterminée, représente le sous système correspondant vu du système dans son ensemble, et réciproquement.

34. Procédé selon la revendication 33, suivant lequel ledit composant commun est l'unique interface entre les la vue supérieure et ladite vue inférieure déterminée.

35. Procédé selon l'une quelconque des revendications précédentes, suivant lequel la phase de modélisation comprend en outre la spécification d'une ou plusieurs métriques de base respectivement associées aux composants.

36. Procédé selon la revendication 35, suivant lequel les métriques de base comprennent, une métrique d'efficacité des parades, une métrique d'efficacité de détection des attaques, et/ou une métrique des moyens d'un attaquant.

37. Procédé selon l'une quelconque des revendications précédentes, suivant lequel la phase de simulation comprend en le calcul d'une ou plusieurs métriques de probabilité de sinistre.

38. Procédé selon la revendication 37, suivant lequel les métriques de probabilité de sinistre comprennent une métrique de probabilité de passage d'une attaque sur un composant.

39. Procédé selon les revendications 36 et 38, suivant lequel la métrique de probabilité de passage d'une attaque sur un composant est calculée suivant la formule " probabilité de passage = (moyens de l'attaquant) / (efficacité de la protection)".

40. Procédé selon la revendication 37, suivant lequel les métriques de probabilité de sinistre comprennent une métrique de probabilité de non détection d'une attaque sur un composant.

41. Procédé selon les revendications 36 et 40, suivant lequel la métrique de probabilité de non détection d'une attaque sur un composant est calculée suivant la formule "probabilité de non détection = (moyens de l'attaquant) / (efficacité de la détection)".

42. Dispositif pour la mise en œuvre d'un procédé selon l'une quelconque des revendications précédentes, comprenant une interface Homme / machine (15) pour la mise en œuvre de la phase de modélisation et/ou un moteur attaques / parades (16) pour la mise en œuvre de la phase de simulation

43. Dispositif selon la revendication 42, dans lequel l'interface Homme / machine présente une fonctionnalité d'affichage en multi vues du système modélisé.

44. Dispositif selon la revendication 42 ou la revendication 43, dans lequel l'interface Homme / machine permet d'afficher le système modélisé selon un modèle composants / relations.

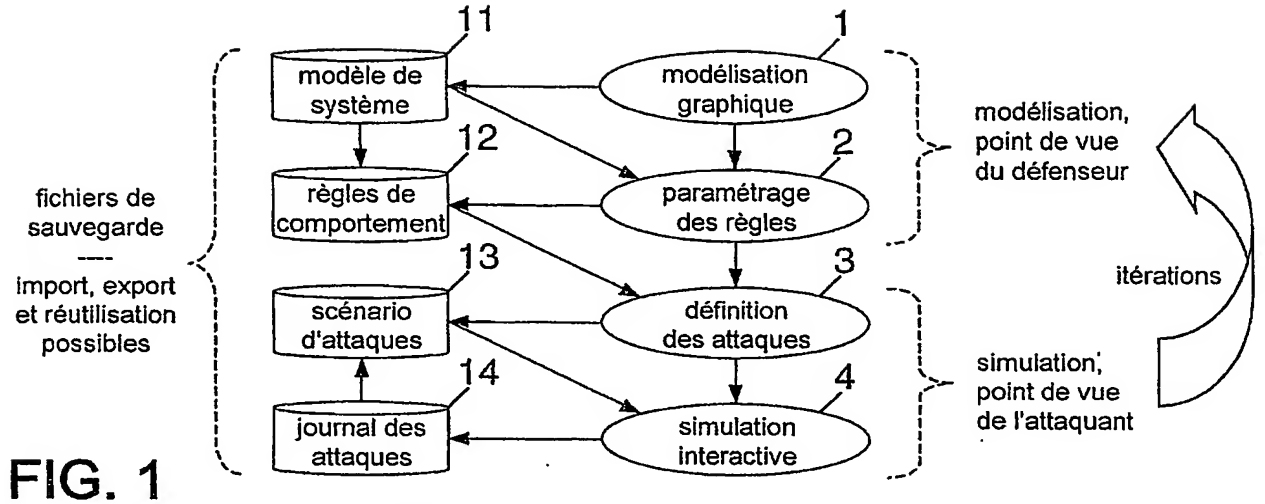


FIG. 1

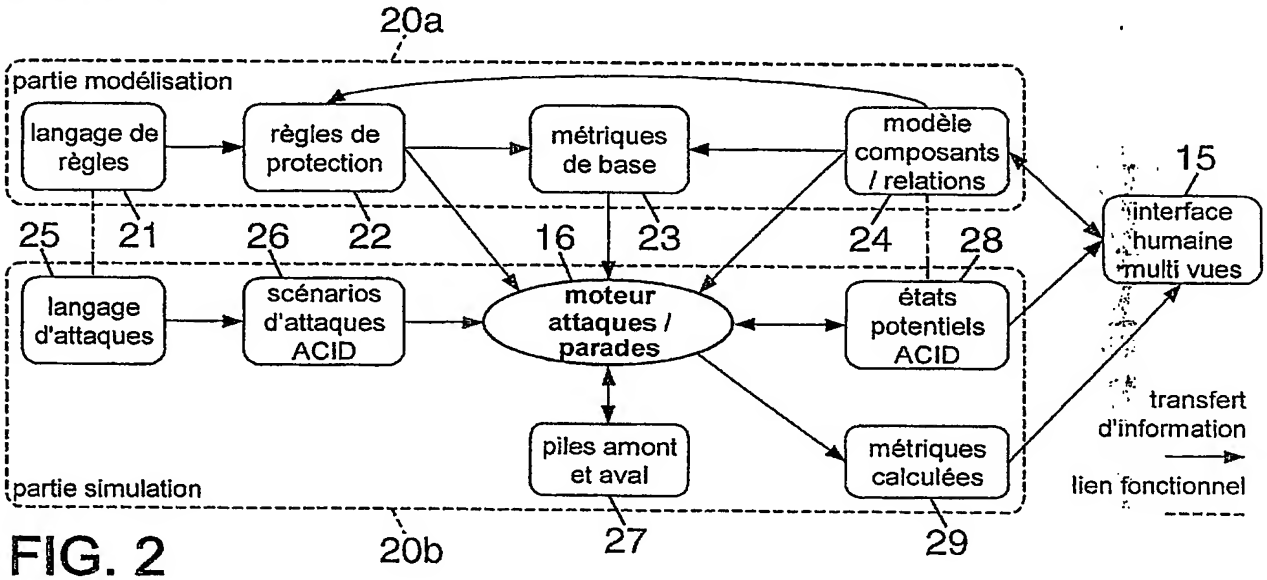


FIG. 2

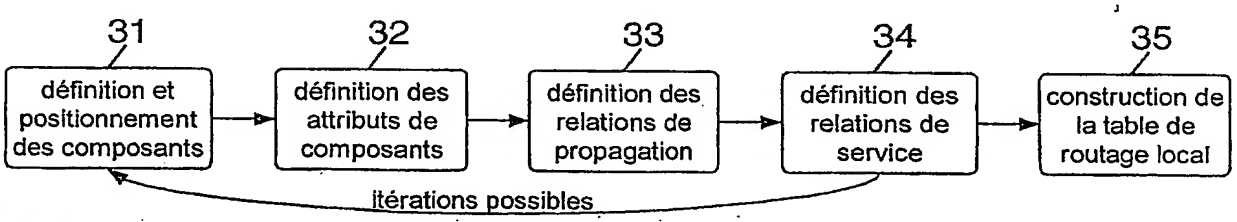


FIG. 3

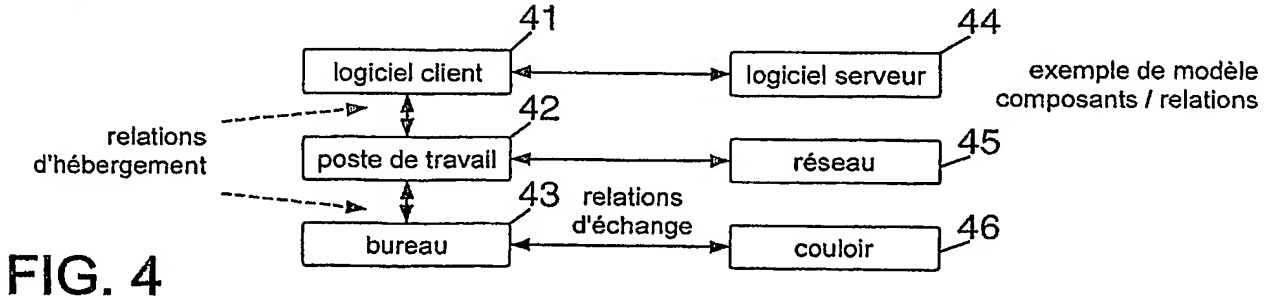


FIG. 4

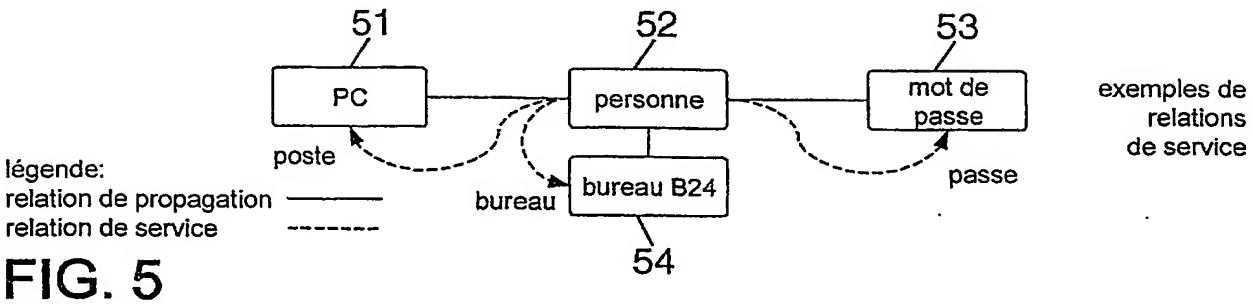


FIG. 5

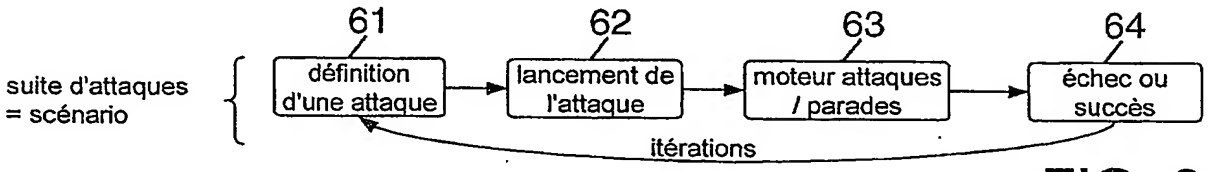
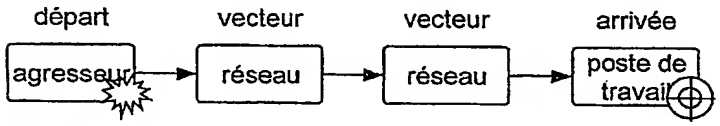
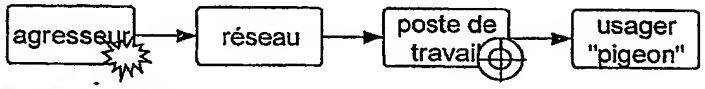


FIG. 6



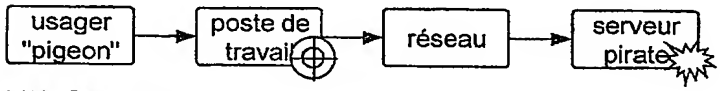
l'agresseur sature le poste de travail par du trafic artificiel transmis via le réseau

FIG. 7a



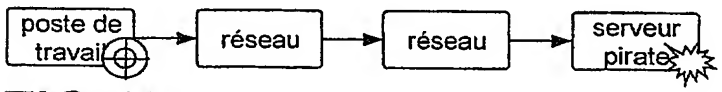
l'agresseur envoie un virus mail exécuté par l'usager "pigeon", ce qui altère le poste

FIG. 7b



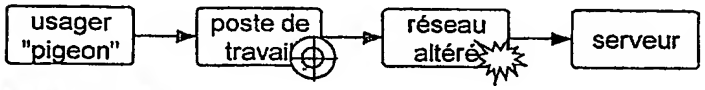
le "pigeon" consulte une page web et reçoit une applet Java pirate qui altère le poste

FIG. 7c



le poste de travail (rendu actif) ouvre une "reverse back door" vers la machine pirate

FIG. 7d



une session passe par un réseau altéré, dévoilant ainsi des informations du poste

FIG. 7e

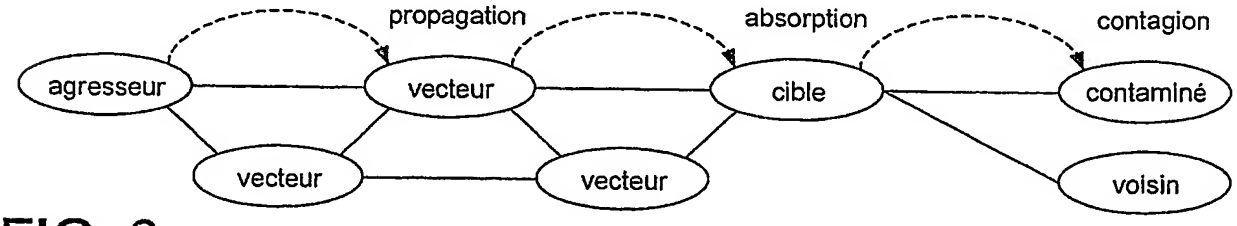


FIG. 8

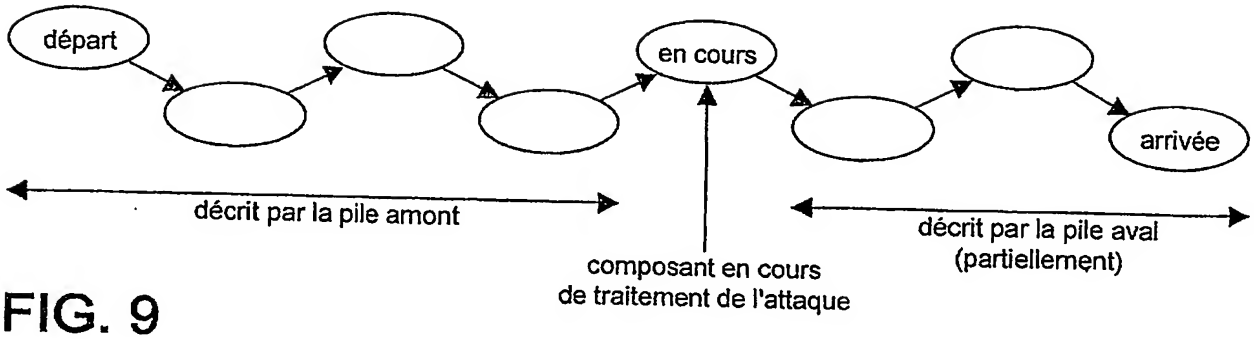


FIG. 9

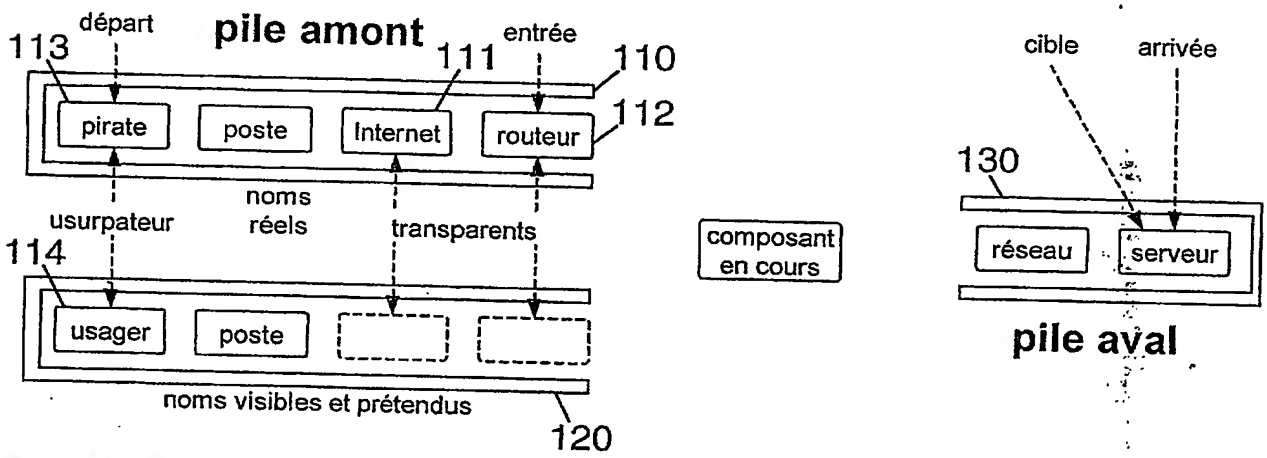


FIG. 10

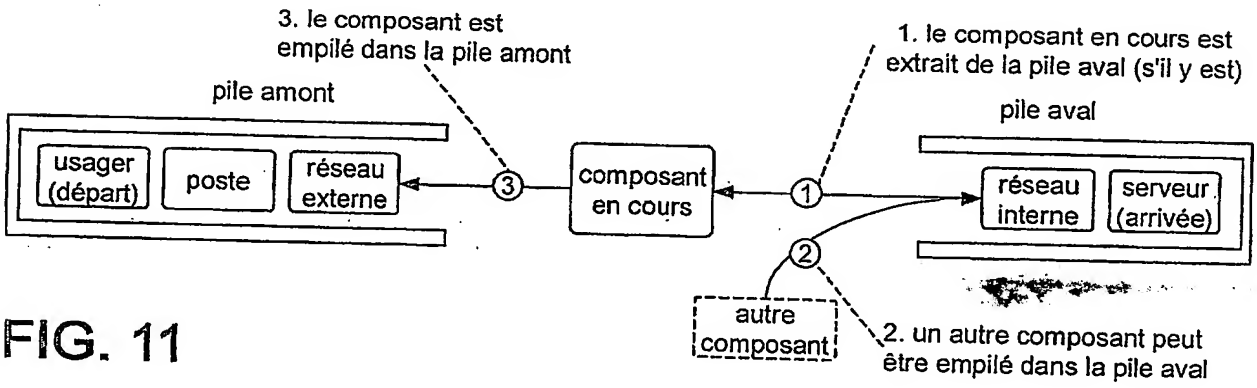


FIG. 11

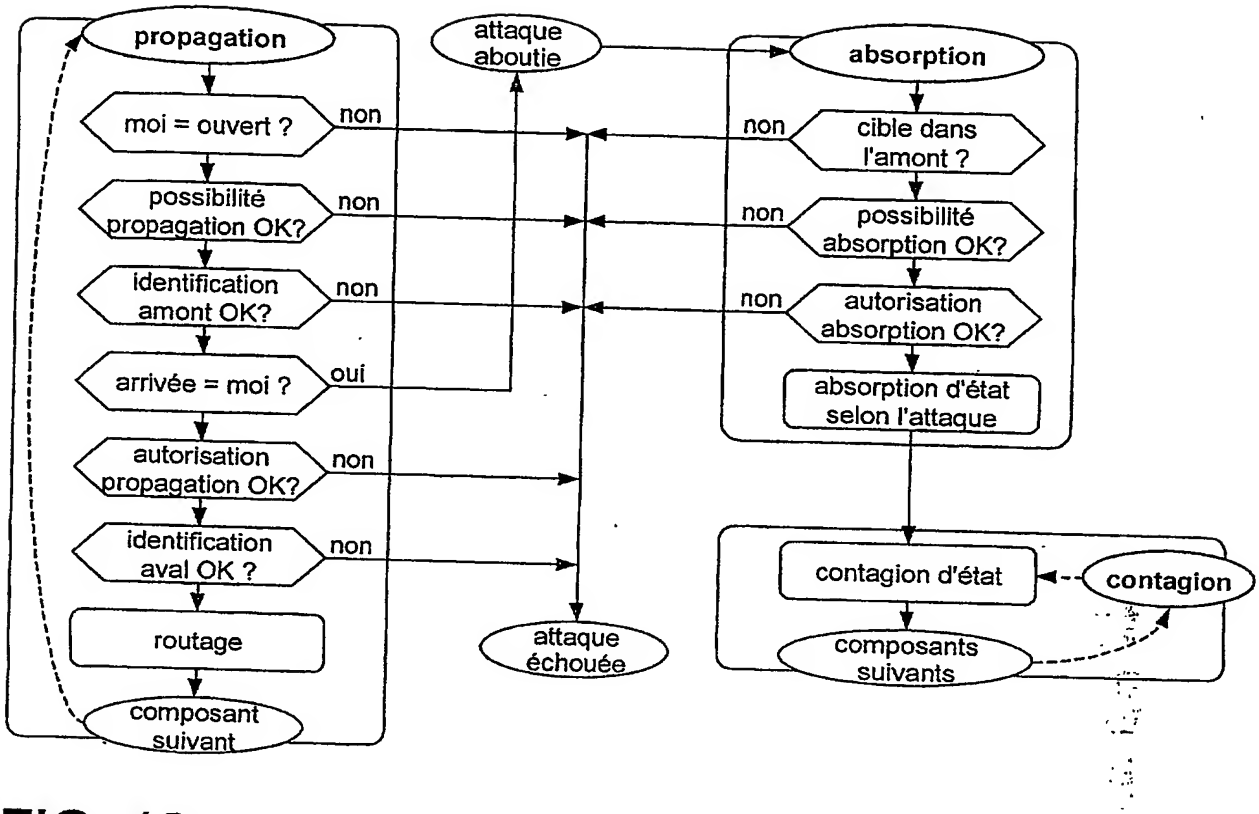

propagation
moi = ouvert ?
non
possibilité propagation OK?
non
identification amont OK?
non
arrivée = moi ?
oui
autorisation propagation OK?
non
identification aval OK ?
non
routage
composant suivant
attaque aboutie
attaque échouée
absorption
non
cible dans l'amont ?
non
possibilité absorption OK?
non
autorisation absorption OK?
absorption d'état selon l'attaque
contagion d'état
contagion
composants suivants

FIG. 12

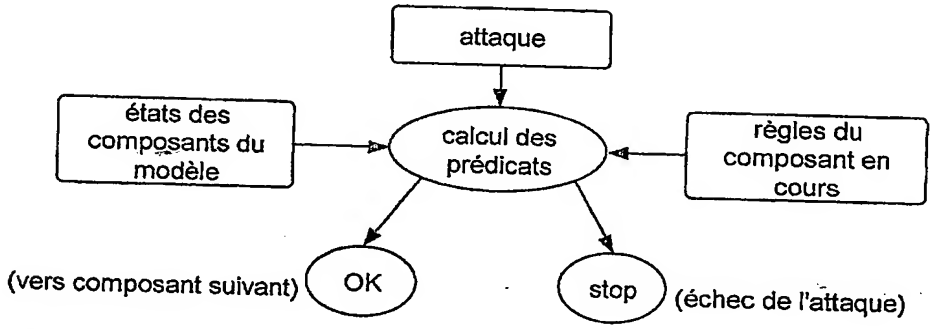

attaque
états des composants du modèle
calcul des prédicats
règles du composant en cours
(vers composant suivant)
OK
stop
(échec de l'attaque)

FIG. 13

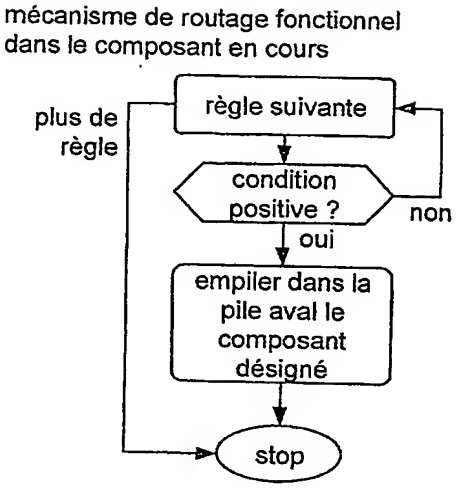


FIG. 14a

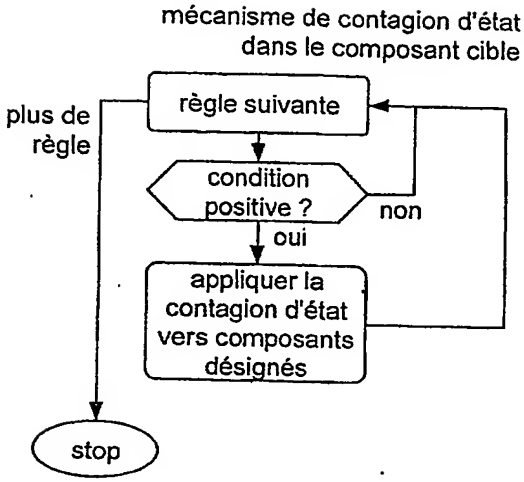


FIG. 14b

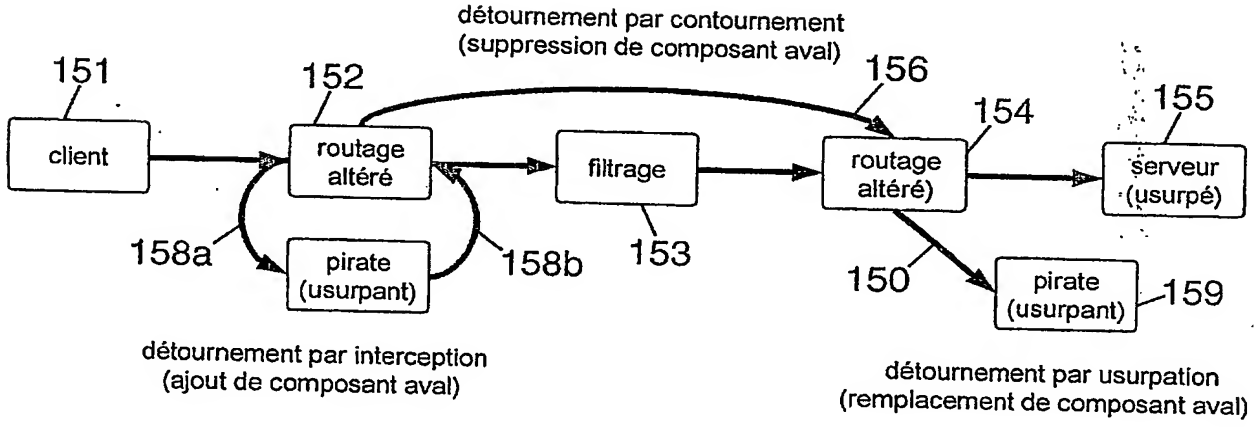


FIG. 15

Exemple de modèle
avec 3 vues

avec chemin
d'attaque sur
les 3 vues.

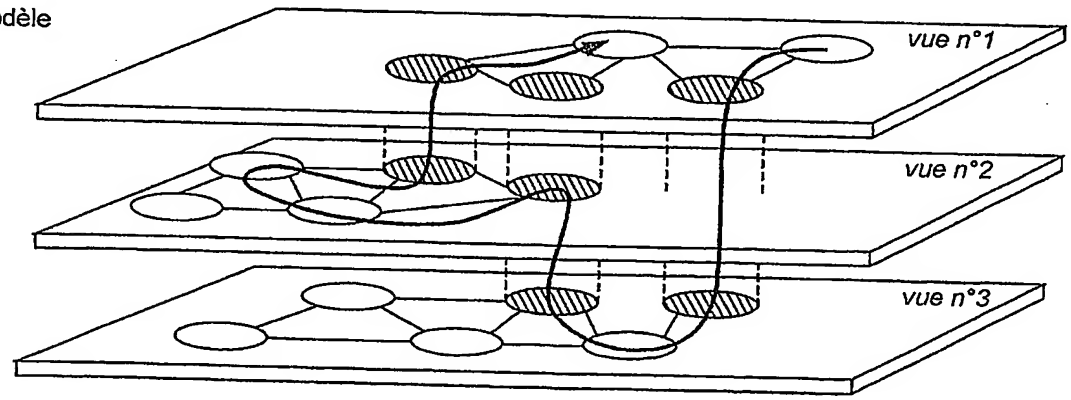


FIG. 16

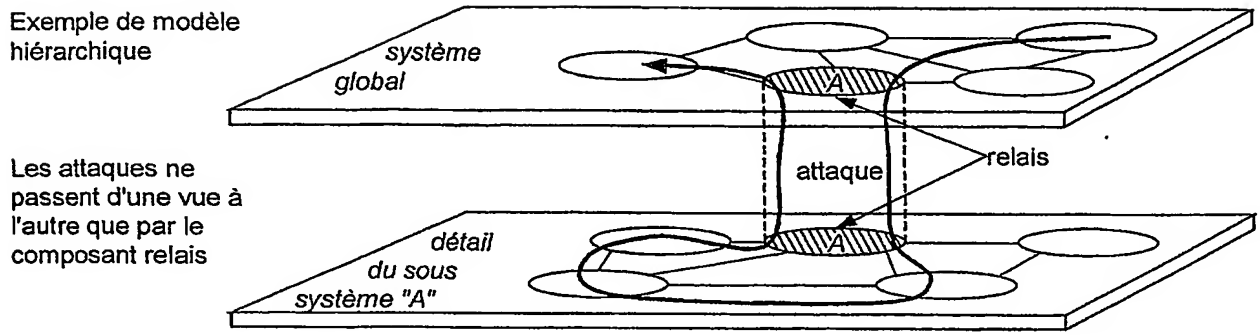

Exemple de modèle hiérarchique
système global
A
attaque
relais
Les attaques ne passent d'une vue à l'autre que par le composant relais
détail du sous système "A"
A

FIG. 17

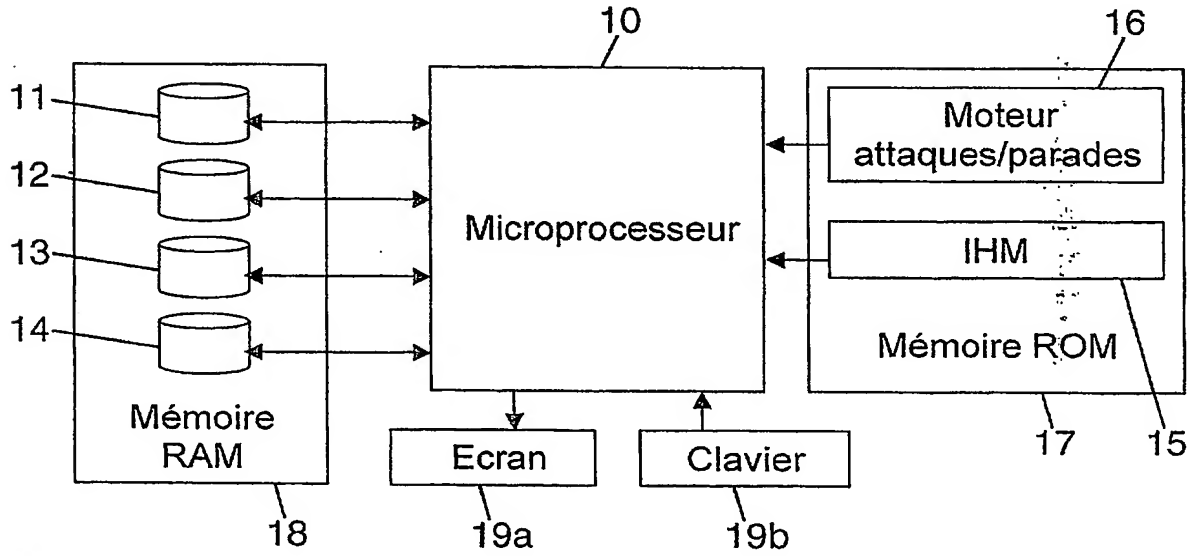

10
16
11
12
13
14
Microprocesseur
Moteur attaques/parades
IHM
Mémoire ROM
Mémoire RAM
Ecran
Clavier
17
15
18
19a
19b

FIG. 18

INPI
INSTITUT NATIONAL DE LA PROPRIETE INDUSTRIELLE

BREVET D'INVENTION
CERTIFICAT D'UTILITÉ

Code de la propriété intellectuelle - Livre VI

cerfa N° 11235*03

DÉPARTEMENT DES BREVETS
26 bis, rue de Saint Pétersbourg
75800 Paris Cedex 08
Téléphone : 33 (1) 53 04 53 04 Télécopie : 33 (1) 42 94 86 54

DÉSIGNATION D'INVENTEUR(S) Page N° 1 / 1

INV

(À fournir dans le cas où les demandeurs et les inventeurs ne sont pas les mêmes personnes)

Cet imprimé est à remplir lisiblement à l'encre noire

DB 113 (W / 270601

Vos références pour ce dossier *(facultatif)*	BFF020343
N° D'ENREGISTREMENT NATIONAL	0214279

TITRE DE L'INVENTION (200 caractères ou espaces maximum)

PROCEDE ET DISPOSITIF D'ANALYSE DE LA SECURITE D'UN SYSTEME D'INFORMATION

LE(S) DEMANDEUR(S) :

EADS DEFENCE AND SECURITY NETWORKS

DESIGNE(NT) EN TANT QU'INVENTEUR(S) :

1 Nom		MAHIEU
Prénoms		Michel
Adresse	Rue	26, rue des Grands Champs
	Code postal et ville	9 1 4 3 0 VAUHALLAN
Société d'appartenance *(facultatif)*		
2 Nom		
Prénoms		
Adresse	Rue	
	Code postal et ville	
Société d'appartenance *(facultatif)*		
3 Nom		
Prénoms		
Adresse	Rue	
	Code postal et ville	
Société d'appartenance *(facultatif)*		

S'il y a plus de trois inventeurs, utilisez plusieurs formulaires. Indiquez en haut à droite le N° de la page suivi du nombre de pages.

DATE ET SIGNATURE(S)
DU (DES) DEMANDEUR(S)
OU DU MANDATAIRE
(Nom et qualité du signataire)

Le 14 novembre 2002
CABINET PLASSERAUD
Stéphane VERDURE
CPI n° 97-0901

La loi n°78-17 du 6 janvier 1978 relative à l'informatique, aux fichiers et aux libertés s'applique aux réponses faites à ce formulaire.
Elle garantit un droit d'accès et de rectification pour les données vous concernant auprès de l'INPI.